Seed design framework for mapping SOLiD reads

Laurent Noé, Marta Gîrdea, and Gregory Kucherov*

INRIA Lille - Nord Europe, LIFL/CNRS, Université Lille 1, 59655 Villeneuve d'Ascq, France

Abstract. The advent of high-throughput sequencing technologies constituted a major advance in genomic studies, offering new prospects in a wide range of applications. We propose a rigorous and flexible algorithmic solution to mapping SOLiD color-space reads to a reference genome. The solution relies on an advanced method of seed design that uses a faithful probabilistic model of read matches and, on the other hand, a novel seeding principle especially adapted to read mapping. Our method can handle both lossy and lossless frameworks and is able to distinguish, at the level of seed design, between SNPs and reading errors. We illustrate our approach by several seed designs and demonstrate their efficiency.

1 Introduction

High-throughput sequencing technologies can produce hundreds of millions of DNA sequence reads in a single run, providing faster and less expensive solutions to a wide range of genomic problems. Among them, the popular SOLiD system (Applied Biosystems) features a 2-base encoding of reads, with an error-correcting capability helping to reduce the error rate and to better distinguish between sequencing errors and SNPs.

In this paper, we propose a rigorous and flexible algorithmic approach to mapping SOLiD color-space reads to a reference genome, capable to take into account various external parameters as well as intrinsic properties of reads resulting from the SOLiD technology. The flexibility and power of our approach comes from an advanced use of *spaced seeds* [1,2].

The main novelty of our method is an *advanced seed design* based on a *faithful probabilistic model of SOLiD read alignments* incorporating reading errors, SNPs and base indels, and, on the other hand, on a *new seeding principle* especially adapted for read mapping. The latter relies on the use of a small number of seeds (in practice, typically two) *designed simultaneously with a set of position on the read where they can hit*. We call this principle *position-restricted seeds*. Advantageously, it allows us to take into account, in a subtle way, read properties such as a non-uniform distribution of reading errors along the read, or a tendency of

* On leave in J.-V.Poncelet Lab, Moscow, Russia

reading errors to occur periodically at a distance of 5 positions, which are observed artifacts of the SOLiD technology.

A number of algorithms and associated software programs for read mapping have been recently published. Several of them such as MAQ [3], MOSAIK [4], MPSCAN [5] PASS [6], PerM [7], RazerS [8], SHRiMP [9] or ZOOM [10] apply contiguous or spaced seeding techniques, requiring one or several hits per read. Other programs approach the problem differently, e.g., by using the Burrows-Wheeler transform (Bowtie [11], BWA [12], SOAP2 [13]), suffix arrays (segemehl [14], BFAST [15]), variations of the Rabin-Karp algorithm (SOCS [16]) or a non-deterministic automata matching algorithm on a keyword tree of the search strings (PatMaN [17]). Some tools, such as segemehl [14] or Eland [18], are designed for 454 and Illumina reads and thus do not deal with the characteristics of the SOLiD encoding which is the subject of this paper. Also, it should be noted that, in many cases, sensitivity is sacrificed in favor of speed: most methods find similarities up to a small number of mismatches, and few approaches account for nucleotide insertions and deletions.

Seed-based methods for read mapping use different seeding strategies. SHRiMP [9] uses spaced seeds that can hit at any position of the read and introduces a lower bound on the number of hits within one read. MAQ [3] uses six light-weight seeds allowed to hit in the initial part of the read. ZOOM [10] proposes to use a small number (4-6) of spaced seeds each applying at a fixed position, to ensure a lossless search with respect to a given number of mismatches. In the lossless framework, PerM [7] proposes to use "periodic seeds" (see also [19]) to save on the index size.

Despite the number of proposed solutions, none of them relies on a systematic seed design method taking into account (other than very empirically) statistical properties of reads. In this paper, we present a seed design based on Hidden Markov models of read matches, using a formal finite automata-based approach previously developed in [20]. To the best of our knowledge, this is the first time that the seed design for read mapping is done based on a rigorous probabilistic modeling.

Our approach allows us to design seeds in both lossy and lossless frameworks. In the lossless framework, where the goal is to detect all read occurrences within a specified number of mismatches, we have the flexibility of partitioning this number into reading errors and SNPs.

As a result, we obtain a very efficient mapping algorithm combining a small number of seeds and therefore a reasonable amount of index memory with guaranteed sensitivity and small running time, due to a restricted subset of positions where seeds should be applied.

2 AB SOLiD reads: encoding and technological artifacts

The SOLiD System [21] enables massively parallel sequencing of clonally amplified DNA fragments. This sequencing technology is based on sequential ligation of dye-labeled oligonucleotide probes, each probe assaying two base positions at a time. The system uses four fluorescent dyes to encode for the sixteen possible 2-base combinations. Consequently, a DNA fragment is represented by the initial base followed by a sequence of overlapping dimers, each encoded with one of four colors using a degenerate coding scheme that satisfies several rules. Thus, although a single color in a read can represent any of four dimers, the overlapping properties of the dimers and the nature of the color code eliminate ambiguities and allow for error-correcting properties.

As our work relies on modeling the error distribution along the reads, we are particularly interested in several aspects of the sequencing technology that influence this distribution. First, since every color of the read encodes two adjacent bases and therefore every base affects two adjacent colors, it follows that any single base mutation results in the change of two adjacent colors in the read. On the other hand, since cycles of five di-nucleotide readings are performed in order to retrieve the sequence (as described in the documentation of Applied Biosystems [21,22]), we expect reading error bias to appear with a periodicity of 5.

To confirm this intuition, we studied the variation of the reading error probability along the read by analyzing statistical properties of about a million of SOLiD reads of the *S. cerevisiae* genome. In this analysis, we used the qualities Q_l associated to each position l on the read, which relate to the error probability p_e^l through $Q_l = -10 \cdot \log_{10}(p_e^l)$ [23].

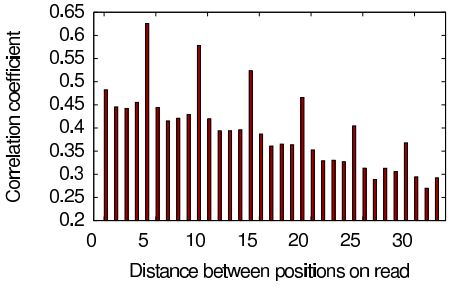

Fig. 1. Position quality correlation coefficient depending on the distance between read positions.

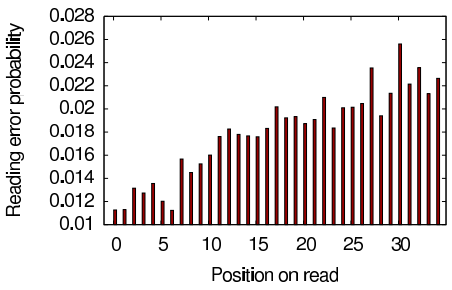

Fig. 2. Average reading error probability at each read position.

We computed the quality correlation between read positions depending on the distance between them. Formally, if m is the read length, then for each $i \in \{1, .., m-1\}$, we computed the correlation through the following standard formula $c(i) = \frac{E((Q_j - \widetilde{Q})(Q_{j+i} - \widetilde{Q}))}{(\sigma_Q)^2}$, where $E(\cdot)$ is the expectation, $\widetilde{Q}$ the average quality along the read, and σ_Q the standard deviation of quality values. The result is given in Figure 1. It shows significantly higher correlations (up to 0.63) between pairs of positions located at distances that are multiples of 5.

Additionally, we studied the behavior of reading error probability values along the read. As shown in Figure 2, the error probability tends to increase towards the end of the read, making the last positions of the color sequence less reliable when searching for similarities.

3 Seed design for mapping SOLiD reads

3.1 Seed design: background

Spaced seeds, first proposed in the context of DNA sequence alignment by the PatternHunter algorithm [1], represent a powerful tool for enhancing the efficiency of the sequence search.

Using a spaced seed instead of a contiguous stretch of identical nucleotides to select a potential similarity region can improve the sensitivity of the search for a given selectivity level [1]. Furthermore, using a seed family, i.e. several seeds simultaneously instead of a single seed, further improves the sensibility/selectivity trade-off [24,25]. The price for using seed families is the necessity to store in memory several indexes, one for each seed. In practice, however, using in the search a small number of seeds can significantly improve the sensitivity/selectivity ratio.

A crucial feature of spaced seeds is their capacity to be adapted to different search situations. Spaced seeds can be *designed* to capture statistical properties of sequences to be searched. For example, [26,27] report on designing spaced seeds adapted to the search of coding regions. One of the contributions of this paper is a rigorous design of seeds adapted to mapping genomic reads issued from the SOLiD technology. Note that here we will work with regular spaced seeds rather than more advanced subset seeds [20,27,28], as there is very little or no information in discriminating among different classes of mismatches that can be used to our advantage.

One has to distinguish between the *lossy* and *lossless* cases of seed-based search. In the lossy case we are allowed to miss a fraction of target matches, and the usual goal of seed design is to maximize the sensitivity

over a class of seeds verifying a certain selectivity level. In the lossless case we must detect all matches verifying a given dissimilarity threshold (expressed in terms of a number of errors or a minimal score), and the goal of seed design is to compute a minimal set of seeds with the best selectivity that still ensures the lossless search. In the context of read mapping for high-throughput sequencing technologies, both lossy [9,3] and lossless [10,7] frameworks have been used.

Our approach to seed design relies on a methodology proposed in our previous work [20], based on the finite automata theory. A central idea is to model the set of target alignments by a *finite-state probability transducer*, which subsumes the Hidden Markov Model commonly used in biosequence analysis. On the other hand, a seed, or a seed family, is modeled by a *seed automaton* for which we proposed an efficient compact construction [29]. Once these two automata have been specified, computing the seed sensitivity can be done efficiently with a dynamic programming algorithm as described in [20]. The seed design is then done by applying our IEDERA software [20,29,30] that uses the above algorithm to explore the space of possible seeds and select most sensitive seeds using a sampling procedure for seeds and respective hit positions and by performing a local optimization on the best candidates.

Here we apply this methodology to seed design for mapping SOLiD reads, both in the lossy and lossless frameworks. Besides, we introduce an important novelty in the definition of seeds, especially advantageous for mapping short reads: *position-restricted seeds*, which are seeds designed together with the set of positions on the read where they can be applied. This can be seen as an intermediate paradigm between applying each seed at every position and the framework of [10] where each seed applies to a designated position of the read. Position-restricted seeds offer an additional power of capturing certain read properties (such as, e.g., an increasing error level towards the end of the read) in a flexible way, without sacrificing the selectivity and thus the speed of the seeding procedure.

3.2 Modeling seeds and SOLiD reads by finite automata

We now present our model of color sequence alignments, built on the observations of Section 2. Note that we consider the reference genome translated into the color alphabet, i.e. both the reads and the genome are represented in color space.

Position-restricted seeds. As shown in Section 2, the reading error probability increases towards the end of the read, implying that a search

for similarity within the last positions of the read could lead to erroneous results or no results at all. Hence, we can improve the seed selectivity by favoring hits at initial positions of the read where matches are more likely to be significant. We then define each seed π *jointly* with a set of positions P to which it is applied on the read.

We use the framework of [20] where a seed π is represented by a deterministic finite automaton $\mathcal{Q}$ over the alignment alphabet $\mathcal{A}$ which is here the binary match/mismatch alphabet. Note that the size of $\mathcal{Q}$ is a crucial parameter in the algorithm of [20] to compute the sensitivity of the seed. An efficient construction of such an automaton has been studied in [29]: it has the optimal size of $(w+1)2^{s-w}$ states, where s and w are respectively the *span* (length) and *weight* (number of *match* symbols) of the seed.

Let m be the read size. To take into account the set of allowed positions, we compute the product of $\mathcal{Q}$ with an automaton λ_P consisting of a linear chain of $m+1$ states $q_0, q_1, \ldots, q_m$, where q_0 is the initial state, and for every q_i, both outgoing transitions lead to q_{i+1}. Final states of the automaton reflect the set of possible positions P where the seed is allowed to hit: a state q_i is final iff $i - s \in P$.

A trivial upper bound on the size of the product automaton for a spaced seed of span s and weight w is $(w+1) \cdot 2^{s-w} \cdot m$. This bound can be improved using the notion of matching prefix, as explained in [29]. Thus, an economical implementation of the product of $\mathcal{Q}$ by λ taking into account the set of matching positions P always produces at most $(w+1) \cdot 2^{s-w} \cdot |P| + m$ states.

Furthermore, consider an interval graph of the possible placements of the seed on the read, where each placement spans over an interval of s positions. The chromatic number c of this graph can be easily computed, providing the maximal number of overlapping seeds. We observe that if this number is small (compared to $(s - w + log(w))$), then the size of the product automaton is bounded by $O((m+1) \cdot 2^c)$.

Model for SNPs and reading errors As explained in Section 2, there are two independent sources of errors in reads with respect to the reference genome: reading errors and SNPs/indels, i.e., *bona fide* differences between the reference genome and sequenced data. We represent each of these sources by a separate Hidden Markov Model (viewed as a probabilistic transducer, see [20]), combined in a model which allows all error types to be cumulated in the resulting sequences.

The **SNP/Indel model**, denoted $M_{SNP/I}$, (Figure 3) has three states: *Match*, *SNP* and *Indel*, referring to matches, mismatches, and indels *at the nucleotide level*, and is parametrized by SNP and Indel occurrence probabilities, denoted p_{SNP} and p_{Indel}. Each transition of $M_{SNP/I}$ generates a *color match, mismatch or indel*, with probabilities p_m^c, p_e^c, and p_i^c respectively, defined as follows. An insertion or deletion of n nucleotides appears at the color level as an insertion/deletion of n colors preceded in 3/4 cases by a color mismatch [21]. Hence, the $p_e^c = 0.75$ when entering the *Indel* state, and $p_i^c = 1$ for any transition having the *Indel* state as source. A nucleotide mutation is reflected in the color encoding by a change of two adjacent colors (and, more generally, n consecutive mutations affect $n + 1$ consecutive colors [21]). Thus, $p_e^c = 1$ when entering or leaving the *SNP* state, and a color match/mismatch mixture when staying in the mismatch state, since color matches may occur inside stretches of consecutive SNPs. Finally, $p_m^c = 1$ when looping on the M state.

The **reading errors** are handled by a more complex model, denoted M_{RE} (Figure 4). Basically, it is composed of several submodels, one for each possible arrangement of reading errors on a cycle of 5 positions. Within these submodels, the transitions shown in red correspond to periodic reading errors, and generate reading errors with a fixed, usually high probability p_{err}. This simulates the periodicity property shown in Figure 1. Switching from one cyclic submodel to another with a higher reading error rate (by adding another red transition, with high error probability) can occur at any moment with a fixed probability p_s.

The transitions shown in black in the model from Figure 4 have an error emission probability of 0. However, in the complete reading error model, we wish to simulate the error probability that increases towards the end (in conformity with Figure 2). We do this by ensuring that reading errors are generated on these transitions with a probability $p'_{err}(pos)$ (lower than p_{err}) given by an increasing function of the current position *pos* on the read. Technically, this is achieved by multiplying the automaton in Figure 4 by a linear automaton with $m + 1$ states, where m is the read length and the i-th transition generates a reading error (color mismatch) with the probability $p'_{err}(i)$. The reading error emission probability in the product model is computed as the maximum of the two reading error probabilities encountered in the multiplied models.

The **final model**, which combines both error sources, is the product of $M_{SNP/I}$ and M_{RE}. While the states and transitions of the product model are defined in the classic manner, the emissions are defined through specific rules based on symbol priorities. If corresponding transitions of

$M_{SNP/I}$ and M_{RE} generate symbols α and β with probabilities p_1 and p_2 respectively, then the product automaton generates the dominant symbol between α and β with probability $p_1 p_2$. Different probabilities obtained in this way for the same symbol are added up.

The dominance relation is defined as follows: *indels* are dominant over both *mismatches* and *matches*, and *mismatches* dominate *matches*. For example, $(indel, mismatch)$ results in an *indel*, $(mismatch, mismatch)$ and $(match, mismatch)$ represent *mismatch*, $(match, match)$ is a *match*. This approach ensures that errors generated by each of the two models are superposed.

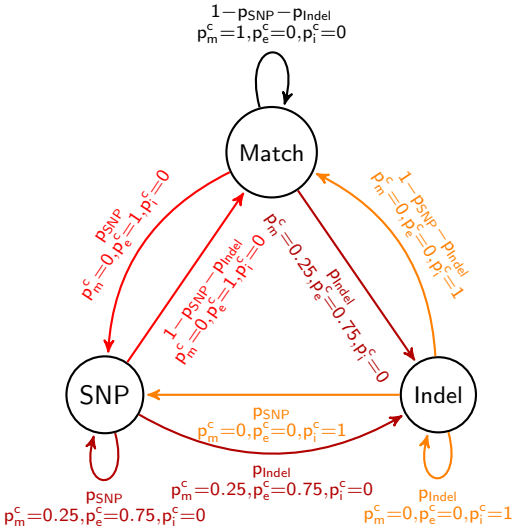

Fig. 3. Model of SNPs and Indels ($M_{SNP/I}$). Colors of transitions correspond to emitted errors: black for color matches, red for mismatches, yellow for indels, and dark red for a mixture of matches (0.25) and mismatches (0.75)

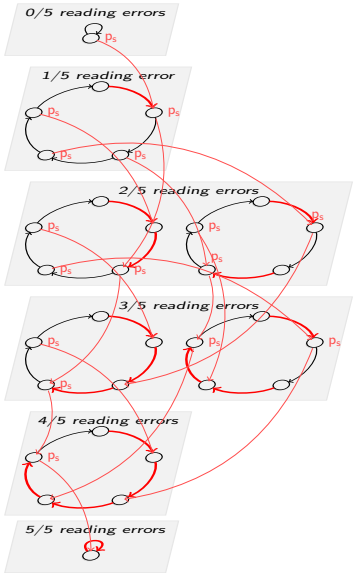

Fig. 4. Reading error automaton

3.3 Computing the sensitivity or testing the lossless property

Given an automaton $\mathcal{Q}$ specifying a family of seeds possibly restricted to a set of positions, we have to compute its sensitivity (in the lossy framework) or to test whether it is lossless (in the lossless framework).

The sensitivity of a seed family is defined [1,31] as the probability for at least one of the seeds to hit a read alignment with respect to a given probabilistic model of the alignment. As outlined in Section 3.1, this is done using the dynamic programming technique of [20]. We therefore omit further details.

In the lossless framework, we have to test if the seed specified by $\mathcal{Q}$ is lossless, i.e. hits *all* the target alignments. The set of target alignments is defined through a threshold number of allowed mismatches.

A straightforward way to test the lossless property of $\mathcal{Q}$ would be to construct a deterministic automaton recognizing the set of all target alignments and then to test if the language of this automaton is included in the language of $\mathcal{Q}$. This, however, is unfeasible in practice. The automaton of all target alignments is much too costly to construct: for example, in the case of threshold of k mismatches, there are $\sum_{a=0}^{k} \binom{m}{a}$ different alignments of length m, and the Aho-Corasick automaton of these strings would have $\sum_{a=0}^{k+1} \binom{m}{a}$ states. Moreover, testing the inclusion would lead to computing the product of this automaton with $\mathcal{Q}$ which would multiply the number of states by that of $\mathcal{Q}$.

Alternatively, we propose an efficient dynamic programming algorithm directly applied to $\mathcal{Q}$ that can verify the inclusion. This algorithm computes, for each state q of $\mathcal{Q}$, and for each iteration $i \in [1..m]$, the minimal number of mismatches needed to reach q at step i. Let k be the threshold for the number of mismatches. Then, the lossless condition holds iff at step m, all non-final states have a number of mismatches greater than k. Indeed, if there is a non-final state that has a number of errors at most k after m steps, then there is at least one string of length m with at most k mismatches that is not detected by the automaton, which contradicts the lossless condition. This algorithm is of time complexity $\mathcal{O}(|\mathcal{Q}|\cdot|\mathcal{A}|\cdot m)$, and space complexity $\mathcal{O}(|\mathcal{Q}|\cdot|\mathcal{A}|)$, where $\mathcal{A}$ is the alphabet of the alignment sequences, in our case $\{0,1\}$

To illustrate the efficiency of this algorithm, consider the case of a single spaced seed of span s and weight w, yielding an automaton with at most $(w+1)\cdot 2^{s-w}$ states [32,20]. On this automaton, our method runs in time $O(wm2^{s-w})$ which brings an improvement by a factor of $\frac{2^w}{w}$ of the general bound $\mathcal{O}(m2^s)$ from [33].

In the context of color sequence mapping, it is interesting to define the lossless property with respect to a *maximal number of allowed mismatches that is split between SNPs and reading errors*. Since, in the color space, a SNP appears as two adjacent color mismatches, having k non-consecutive

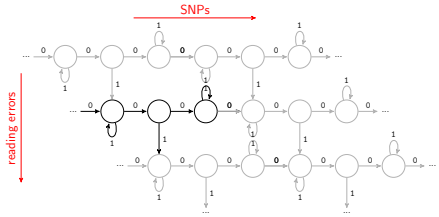

Fig. 5. Building an automaton for k SNPs and h color mismatches from a repeated 3-state pattern.

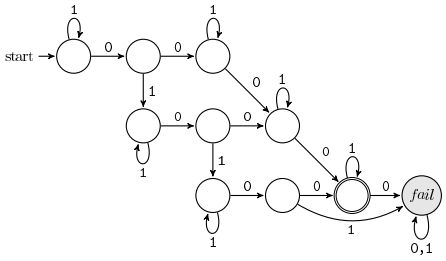

Fig. 6. 1 SNP & 2 errors automaton.

SNPs and h color mismatches implies the possibility to accept $2k + h$ mismatches with the additional restriction that there exist at least k pairs of adjacent ones. The automaton that recognizes the set of alignments verifying this condition on mismatches can be obtained by combining simple 3-state building blocks as depicted in Figure 5. An example of such an automaton, accepting 1 SNP and 2 reading errors, is illustrated in Figure 6 (1 and 0 denote match and mismatch respectively).

Note that the case of consecutive SNPs, resulting in sequences of adjacent color mismatches, is a simpler problem (since consecutive SNPs produce require less mismatches in the color representation than the same number of non-consecutive SNPs) and is covered by the proposed model: a seed that is lossless for alignments with non-consecutive SNPs will also be lossless for alignments with the same number of consecutive SNPs.

To verify the lossless property for k SNPs and h color mismatches, we intersect the corresponding automaton with the seed automaton (thus restricting the set of alignments recognized by the seed to those with k SNPs and h color mismatches) and submit the result to the dynamic programming algorithm of Section 3.3.

4 Experiments and Discussion

We present now several efficient seed designs illustrating our methodology (more examples at `http://bioinfo.lifl.fr/yass/iedera_solid`).

We first computed several sets of seeds of weight 10, restricted to either 10 or 12 positions among the 34 positions of SOLiD reads, each including one or two seeds. Figure 7 shows some of the resulting seeds, together with the corresponding sensitivity values, computed through the methods described in Section 3.

Interestingly, both single seeds 1-Lossy-10p and 1-Lossy-12p contain a double gap, which may reflect that an SNP modifies two adjacent

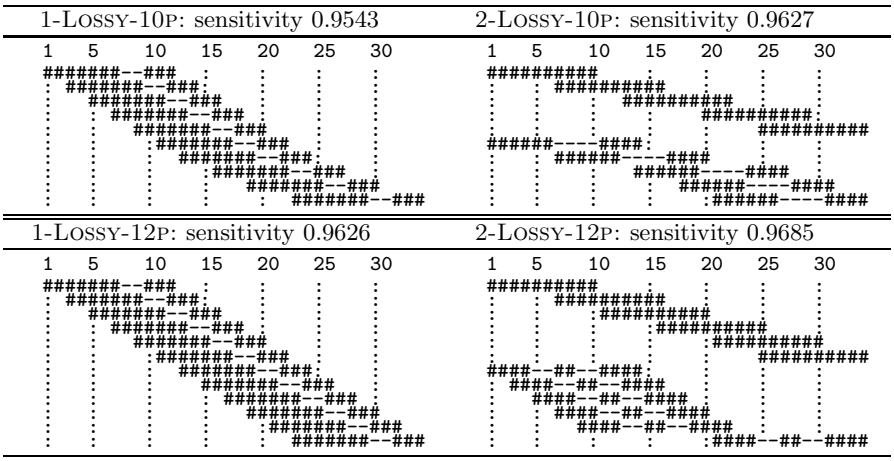

Fig. 7. Position-restricted seeds for 10 (above) and 12 (below) allowed positions. Different placements of a seed correspond to the allowed positions.

colors. However, this gap is not centered but rather shifted at the two-third of the seed (as observed for the best single seeds of [19]). Note also that in the two-seed families 2-Lossy-10p and 2-Lossy-12p, one of the chosen seeds is ungapped. This may be a consequence of the fact that we consider indels in our lossy model, which usually forces the seeds to have a smaller span. Another interesting observation is that two-seed families 2-Lossy-10p and 2-Lossy-12p are actually lossless for the threshold of 3 mismatches, whereas single seeds 1-Lossy-10p and 1-Lossy-12p are not lossless for this setting.

We then focused on the lossless case where the maximal number of allowed mismatches is split between SNPs and reading errors. Using the procedure described in Section 3.3, we computed lossless single and double seeds for one SNP and two reading errors. Results are shown in Figure 8.

Note that the seed 1-Lossless-14p is one of several single seeds of weight 10 we found that satisfied this lossless condition, with no restriction on allowed positions. Interestingly, they all have a very large span (21) and a regular pattern with a periodic structure that can be obtained by iterating a simpler pattern solving the lossless problem for an appropriate *cyclic problem*, following the property we previously described in [19]. For two-seed families, Figure 8 shows a lossless pair of seeds 2-Lossless-8p for read length 33 (which then remains lossless for larger lengths), where each seed is restricted to apply to four positions only.

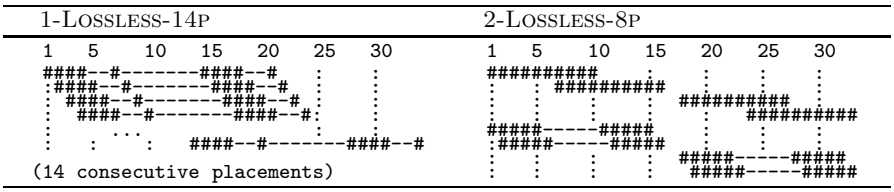

Fig. 8. Lossless position-restricted seeds for 1 SNP and 2 reading errors

To get a better idea of the sensitivity of the obtained seeds applied to real data, we tested them on 100000 reads of length 34 from *S. cerevisiae* and computed the number of read/reference alignments hit by each (single or double) seed. Alignments were defined through the score varying from 28 to 34, under the scoring scheme +1 for match, 0 for color mismatch or SNP, -2 for gaps. Results are presented in Figure 9. One conclusion we can draw is that the performance of lossless seeds 1-Lossless-14p and 2-Lossless-8p decreases quite fast when the alignment score goes down, compared to lossy seeds. Intuitively, this is, in a sense, a price to pay for the lossless condition which usually makes these seeds less appropriate for the alignments with a number of errors exceeding the threshold. Another conclusion is that, as expected, single seeds perform worse than double seeds, although the overall number of positions where seeds apply is the same for both single and double seeds.

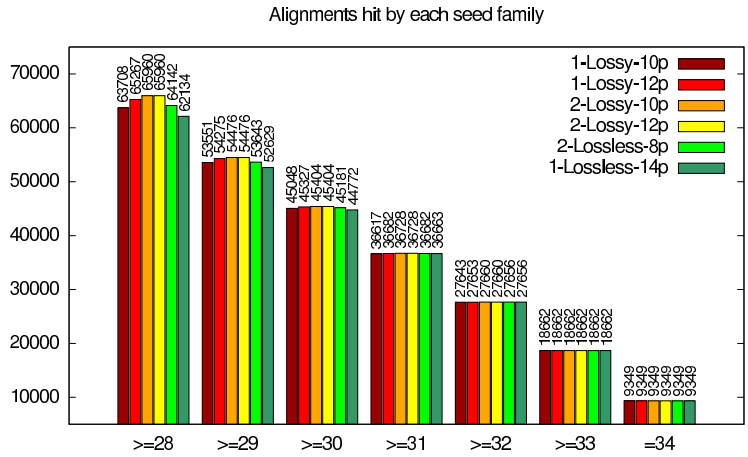

Fig. 9. Number of read alignments with scores between 28 and 34 hit by each seed

Note finally that the choice of the best seed can be affected, on one hand, by different properties of the class of target alignments (number, type and distribution of mismatches and indels etc.) and, on the other hand, by the size of the data and the available computational resources. The former can be captured by our probabilistic models described in Section 3. The latter is related to the choice of the selectivity level, directly affecting the speed of the search, which is defined by the seed weight and the number of allowed positions. Depending on the chosen selectivity, different seeds can (and should) be preferred. Note in this regard that seeds appearing in Figure 9 have different selectivity and are then incomparable *stricto sensu*. A comparison of different seeds for SOLiD read mapping in typical practical situations will be a subject of a separate work.

5 Conclusions and perspectives

In this paper, we presented a seed design framework for mapping SOLiD reads to a reference genomic sequence. Our contributions include the concept of position-restricted seeds, particularly suitable for short alignments with non-uniform error distribution; a model that captures the statistical characteristics of the SOLiD reads, used for the evaluation of lossy seeds; an efficient dynamic programming algorithm for verifying the lossless property of seeds; the ability to distinguish between SNPs and reading errors in seed design.

Our further work will include a more rigorous training of our models and in particular a more accurate estimation of involved probabilities, possibly using advanced methods of assessing the fit of a model. Another interesting question to study is the design of efficient combined lossy/lossless seeds which provide a guarantee to hit all the alignments with a specified number of errors and still have a good sensitivity when this threshold is exceeded. Computing such seeds, however, could be difficult or even unfeasible: for example, lossless seeds tend to have a regular structure (see [19]) while best lossy seeds often have asymmetric and irregular structure. Finally, we want to define and study a lossless property that incorporates possible indels and not only mismatches (SNPs or reading errors) occurring in read alignments.

Acknowledgments

The authors would like to thank Valentina Boeva and Emmanuel Barillot from the *Institut Marie Curie* at Paris for helpful discussions and for

providing the dataset of *Saccharomyces cerevisiae* reads that we used as a testset in our study. We also thank Martin Figeac (*Institut national de la santé et de la recherche médicale*) for sharing insightful knowledge about the SOLiD technology. Laurent Noé was supported by the ANR project CoCoGen (BLAN07-1 185484).

References

1. Ma, B., Tromp, J., Li, M.: PatternHunter: Faster and more sensitive homology search. Bioinformatics **18**(3) (2002) 440–445
2. Noé, L., Kucherov, G.: YASS: enhancing the sensitivity of DNA similarity search. Nucleic Acids Research **33**(Web Server Issue) (2005) W540–W543
3. Li, H., Ruan, J., Durbin, R.: Mapping short DNA sequencing reads and calling variants using mapping quality scores. Genome Research **18** (2008) 1851–1858
4. Strömberg, M., Lee, W.P.: MOSAIK read alignment and assembly program. http://bioinformatics.bc.edu/marthlab/Mosaik (2009)
5. Rivals, E., Salmela, L., Kiiskinen, P., Kalsi, P., Tarhio, J.: MPSCAN: Fast localisation of multiple reads in genomes. In: Proceedings of the 9th international Workshop on Algorithms in Bioinformatics (WABI). Volume 5724 of LNCS., Springer (2009) 246–260
6. Campagna, D., Albiero, A., Bilardi, A., Caniato, E., Forcato, C., Manavski, S., Vitulo, N., Valle, G.: PASS: a program to align short sequences. Bioinformatics **25**(7) (2009) 967–968
7. Chen, Y., Souaiaia, T., Chen, T.: PerM: Efficient mapping of short sequencing reads with periodic full sensitive spaced seeds. Bioinformatics **25**(19) (2009) 2514–2521
8. Weese, D., Emde, A., Rausch, T., Döring, A., Reinert, K.: RazerS–fast read mapping with sensitivity control. Genome Research **19**(9) (2009) 1646–1654
9. Rumble, S.M., Lacroute, P., Dalca, A.V., Fiume, M., Sidow, A., Brudno, M.: SHRiMP: Accurate mapping of short color-space reads. PLoS Comp. Biol **5**(5) (2009)
10. Lin, H., Zhang, Z., Zhang, M., Ma, B., Li, M.: ZOOM! zillions of oligos mapped. Bioinformatics **24**(21) (2008) 2431–2437
11. Langmead, B., Trapnell, C., Pop, M., Salzberg, S.: Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biology **10**(3) (2009)
12. Li, H., Durbin, R.: Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics **25**(14) (2009) 1754–1760
13. Li, R., Yu, C., Li, Y., Lam, T., Yiu, S., Kristiansen, K., Wang, J.: SOAP2: an improved ultrafast tool for short read alignment. Bioinformatics **25**(15) (2009) 1966–1967
14. Hoffmann, S., Otto, C., Kurtz, S., Sharma, C., Khaitovich, P., Stadler, P., Hackermuller, J.: Fast mapping of short sequences with mismatches, insertions and deletions using index structures. PLoS Comp. Biol **5**(9) (2009)
15. Homer, N., Merriman, B., Nelson, S.F.: BFAST: an alignment tool for large scale genome resequencing. PLoS ONE **4**(11) (2009)
16. Ondov, B., Varadarajan, A., Passalacqua, K., Bergman, N.: Efficient mapping of Applied Biosystems SOLiD sequence data to a reference genome for functional genomic applications. Bioinformatics **24**(23) (2008) 2776–2777

17. Prufer, K., Stenzel, U., Dannemann, M., Green, R., Lachmann, M., Kelso, J.: PatMaN: rapid alignment of short sequences to large databases. Bioinformatics **24**(13) (2008) 1530–1531
18. Bentley, D., Balasubramanian, S., Swerdlow, H., Smith, G., Milton, J., Brown, C., Hall, K., Evers, D., Barnes, C., Bignell, H., et al.: Accurate whole human genome sequencing using reversible terminator chemistry. Nature **456**(7218) (2008) 53–59
19. Kucherov, G., Noé, L., Roytberg, M.: Multiseed lossless filtration. IEEE Transactions on Computational Biology and Bioinformatics **2**(1) (2005) 51–61
20. Kucherov, G., Noé, L., Roytberg, M.: A unifying framework for seed sensitivity and its application to subset seeds. J Bioinform Comput Biol **4**(2) (2006) 553–570
21. ABI: A theoretical understanding of 2 base color codes and its application to annotation, error detection, and error correction. methods for annotating 2 base color encoded reads in the SOLiD$^{\text{TM}}$system (2008)
22. ABI: The SOLiD$^{\text{TM}}$3 system. enabling the Next Generation of Science (2009)
23. Ewing, B., Green, P.: Base-calling of automated sequencer traces using phred. II. error probabilities. Genome Research **8**(3) (1998) 186–194
24. Li, M., Ma, B., Kisman, D., Tromp, J.: PatternHunter II: Highly sensitive and fast homology search. J Bioinform Comput Biol **2**(3) (2004) 417–439
25. Sun, Y., Buhler, J.: Designing multiple simultaneous seeds for DNA similarity search. Journal of Computational Biology **12**(6) (2005) 847–861
26. Brejová, B., Brown, D.G., Vinar, T.: Optimal spaced seeds for Hidden Markov Models, with application to homologous coding regions. In: Proceedings of the 14th Symposium on Combinatorial Pattern Matching (CPM). Volume 2676 of LNCS., Springer (2003) 42–54
27. Zhou, L., Stanton, J., Florea, L.: Universal seeds for cDNA-to-genome comparison. BMC Bioinformatics **9**(36) (2008)
28. Yang, J., Zhang, L.: Run probabilities of seed-like patterns and identifying good transition seeds. Journal of Computational Biology **15**(10) (2008) 1295–1313
29. Kucherov, G., Noé, L., Roytberg, M.: Subset seed automaton. In: Proceedings of the 12th International Conference on Implementation and Application of Automata (CIAA). Volume 4783 of LNCS., Springer (2007) 180–191
30. Kucherov, G., Noé, L., Roytberg, M.: Iedera: subset seed design tool. http://bioinfo.lifl.fr/yass/iedera (2009)
31. Keich, U., Li, M., Ma, B., Tromp, J.: On spaced seeds for similarity search. Discrete Applied Mathematics **138**(3) (2004) 253–263 (preliminary version in 2002).
32. Buhler, J., Keich, U., Sun, Y.: Designing seeds for similarity search in genomic DNA. In: Proceedings of the 7th Annual International Conference on Computational Molecular Biology (RECOMB), ACM Press (2003) 67–75
33. Burkhardt, S., Kärkkäinen, J.: Better filtering with gapped q-grams. Fundamenta Informaticae **56**(1,2) (2003) 51–70 Preliminary version in CPM 2001.